



# MKPM: Multi keyword-pair matching for natural language sentences

Xin Lu<sup>1</sup> · Yao Deng<sup>1</sup> · Ting Sun<sup>1</sup> · Yi Gao<sup>1</sup> · Jun Feng<sup>1</sup> · Xia Sun<sup>1</sup> · Richard Sutcliffe<sup>1,2</sup>

Accepted: 2 March 2021  
© The Author(s) 2021

## Abstract

Sentence matching is widely used in various natural language tasks, such as natural language inference, paraphrase identification and question answering. For these tasks, we need to understand the logical and semantic relationship between two sentences. Most current methods use all information within a sentence to build a model and hence determine its relationship to another sentence. However, the information contained in some sentences may cause redundancy or introduce noise, impeding the performance of the model. Therefore, we propose a sentence matching method based on multi keyword-pair matching (MKPM), which uses keyword pairs in two sentences to represent the semantic relationship between them, avoiding the interference of redundancy and noise. Specifically, we first propose a sentence-pair-based attention mechanism *sp-attention* to select the most important word pair from the two sentences as a keyword pair, and then propose a Bi-task architecture to model the semantic information of these keyword pairs. The Bi-task architecture is as follows: 1. In order to understand the semantic relationship at the word level between two sentences, we design a word-pair task (WP-Task), which uses these keyword pairs to complete sentence matching independently. 2. We design a sentence-pair task (SP-Task) to understand the sentence level semantic relationship between the two sentences by sentence denoising. Through the integration of the two tasks, our model can understand sentences more accurately from the two granularities of word and sentence. Experimental results show that our model can achieve state-of-the-art performance in several tasks. Our source code is publicly available<sup>1</sup>.

**Keywords** Sentence matching · Multi keyword-pair · Bi-task architecture

## 1 Introduction

Natural Language Sentence Matching (NLSM), which aims to predict the relationship between two sentences, is a fundamental technique for many natural language processing tasks [1]. For example, in paraphrase identification, it is used to identify whether two sentences express the same intention or semantics [2]. In natural language inference, sentence matching is used to determine whether a hypothetical sentence can be inferred from a given premise sentence [3]. In question answering, it is used to judge whether a

given response can answer a question correctly [4, 5]. However, in these tasks, it is not easy to correctly predict the relationship between two sentences [6], due to the diversity of language expression and the complexity of sentence semantics.

With the development of deep learning, there have been many successful applications to natural language processing tasks [7–10]. For NLSM tasks, there are two popular deep learning methods, sentence-encoding based methods and joint-feature based methods [11]. Because a sentence-encoding based method cannot capture the interactive features between two sentences, people usually take advantage of the joint-feature based method, which utilizes the interactive features or attention information across two sentences to encode them, resulting in a significant improvement in model performance. Attention mechanisms play an important role in sentence alignment and modeling of dependency relationships [12]. The attention mechanism is usually applied to determine the semantic interaction and degree of dependence between two sentences, so that the model can understand them better. In recent years, it has

<https://github.com/Rush1n/MKPM>

✉ Xia Sun  
raindy@nwu.edu.cn

✉ Richard Sutcliffe  
rsutcl@nwu.edu.cn; rsutcl@essex.ac.uk

Extended author information available on the last page of the article.

been found that using a deeper model structure can improve the performance of the model. This kind of structure can extract the deeper semantic features and dependency relationships of sentences [11].

At present, most sentence matching methods predict the semantic relationship between two sentences by comparing all their parts. However, social media posts or web page blogs typically consist of short informal texts containing many colloquialisms: Sentences may contain a lot of redundant information or noise, which affects the understanding of their semantics by a model, thus limiting its performance. On the other hand, we find that some word pairs between two sentences can capture the key semantic relationships without being affected by such noise. For example, here are two sentences in LCQMC [13]: “s1: 冬天去北京可以玩什么 (En): What can you do in Beijing in winter” and “s2: 北京冬天穿什么 (En): What should I wear in winter in Beijing”. We take 3 word pairs: “(北京, 北京) En:(Beijing, Beijing)”, “(冬天, 冬天) En:(winter, winter)”, “(玩, 穿) En:(do, wear)”. Through “(Beijing, Beijing)”, we know that the place relations of the two sentences are the same; through “(winter, winter)”, we know that the time relations of the two sentences are the same; through “(do, wear)”, we know that the intention of these two sentences is different. Therefore, through these three word pairs, we can determine whether the time, place and intention of these two sentences are the same, and according to these relations, we can establish that these two sentences have different semantic relations. If the sentence s2 is changed to “北京冬天玩什么,谁知道 (En): Who knows what to do in winter in Beijing?”, and the third word pair becomes “(do, do)”, it can be determined that the two sentences express the same intention, and we can mark them as having the same semantic relationship. However, there is no “who knows” in s1, but there is in s2. If all the information in the sentences is used, it may mislead the model to judge their relationship, so we think they are noise words. We argue that “what” and “in” are redundant words, which have little effect on the prediction of semantic relationships between two sentences. Therefore, we propose an approach based on multi keyword-pair matching (MKPM) to achieve the sentence matching task.

Given two sentences, we first perform word segmentation, then obtain their embedded representations, and thirdly use the 2-layer Bi-directional Long Short-Term Memory Network (BiLSTM) to encode each word in the sequence to obtain their contextual representation. Next, in the core part of our model, we use the *sp-attention* to select the most important word pair from the two sentences as a keyword pair. In order to capture the semantic relationship between two sentences at different semantic levels, we extract

multiple keyword pairs. We propose a Bi-task architecture to model the semantic information of these keyword pairs, and the specific structure is as follows: 1. The purpose of the word-pair task (WP-Task) is to understand semantic relationships at the word level. Because each keyword pair can complete the sentence matching task independently, the model will learn to identify the keyword pairs containing significant sentence semantic relationships. 2. The purpose of the sentence-pair task (SP-Task) is to use the information from keyword pairs to reduce the noise within sentences in order to represent the key semantics of each sentence, and then to use the interaction features between two sentences to understand semantic relationships at the sentence level. Bi-task integrates WP-Task and SP-Task to predict sentence semantic relationships.

The main contributions of this paper are as follows:

- We propose a word-pair-based attention mechanism *sp-attention* to extract the most important word pairs in sentence pairs, so that the model can more accurately select keyword pairs and better understand the semantic relationship between two sentences.
- We propose a Bi-task architecture to enhance the performance of the model. We use WP-Task to understand the semantic relationships of keyword pairs, and SP-task to understand the semantic relationships of sentence pairs. Through the integration of the two tasks, our model can understand the relationship between two sentences from dual granularities: word level and sentence level.
- We propose a sentence representation method based on keyword pairs to carry out the sentence matching task. We only use keywords that are useful for sentence matching to represent the key semantics of each sentence, thus avoiding interference caused by redundant information.

## 2 Related work

Earlier sentence matching methods mainly rely on conventional methods, such as bag-of-word retrieval, semantic analysis, and syntactic analysis [14–17], but these methods usually have low accuracy and are only suitable for specific tasks.

With the release of datasets such as LCQMC [13], BQ [18], Quora<sup>1</sup> and SNLI [19] and the development of neural networks such as CNN, LSTM and GRU [20–22], more and more attention has been paid to sentence matching. Two methods are widely used, one based on sentence encoding and the other based on joint features. For the sentence-encoding

<sup>1</sup><https://data.quora.com/First-Quora-Dataset-Release-QuestionPairs>

based method, two sentences are first encoded into fixed dimension vectors, and then the two vectors are matched to achieve the relationship prediction [23–27]. Mueller et al. [23] use word embedding with synonym information to encode sentences into vectors of fixed dimensions with the same LSTM, and then use Manhattan distance to measure their similarity. Nie et al. [25] use a stack-based bi-directional LSTM to encode sentences into vectors of fixed dimensions, and then use a classifier over the vector combination to complete the sentence relationship classification. However, these methods only match the two sentence vectors, and the fine-grained information between the two sentences is not captured, resulting in lower performance. As a result, researchers have proposed the joint-feature based method, using the interactive features or attention information between two sentences in the process of encoding sentences at the same time, thus significantly improving the performance of the model [1, 28, 29]. Wang et al. [1] propose a bilateral multi-perspective matching model, which matches sentences in two directions and from four different perspectives, aggregates the matching results into feature vectors with a BiLSTM, and then classifies the feature vectors. Attention mechanisms play an important role in sentence alignment and modeling of semantic relationships [12, 26, 27, 29, 30]. Yin et al. [12] propose an attention-based convolutional network, using three attention schemes to obtain different semantic interactions between sentences so that the model can better understand the sentences. Shen et al. [26] integrate both soft and hard attention into one context fusion model (reinforced self-attention); this efficiently extracts the sparse dependencies between each pair of selected tokens.

Recently, it has been found that using a deeper model structure can improve performance, although this method increases the number of parameters. Gong et al. [11] propose a densely interactive inference network. It achieves a high-level understanding of sentence pairs by extracting semantic features hierarchically from the interaction space. Kim et al. [29] propose a densely-connected co-attentive recurrent neural network, each layer using the attention features and hidden features of all previous recursive layers to achieve a more precise understanding. Most current methods use all the information in a sentence directly, without considering which information is important. So Zhang et al. [31] propose a dynamic re-read network approach; this pays close attention to the important content of the sentence in each step, by re-reading important words to better understand the semantics.

Our work achieves a precise understanding of the semantic relationship between two sentences via a joint-feature based method. We first apply our *sp-attention* mechanism to extract multiple keyword pairs and then use

the information of these keyword pairs to model sentences from two granularities of word and sentence.

## 3 Method

### 3.1 Task definition

We define the sentence matching task as follows: Consider two sentence sequences  $P = (p_1, p_2, \dots, p_M)$  and  $Q = (q_1, q_2, \dots, q_N)$ , where  $p$  and  $q$  are the tokens of sentences  $P$  and  $Q$  respectively,  $M$  and  $N$  are the lengths of sentences  $P$  and  $Q$  respectively. The goal is to train a classifier so that it can correctly predict the relationship between  $P$  and  $Q$ . We propose a sentence matching method based on multi keyword-pair matching (MKPM) to achieve sentence matching so as to understand the semantic relationship between two sentences accurately.

### 3.2 Overall architecture of MKPM

The overall architecture of MPKM is shown in Fig. 1. MPKM comprises the Word Representation Layer, the Keyword Pair Extraction (KWPE) Layer, the Denoising Layer, the Matching Layer and the Prediction Layer. The purpose of the Word Representation Layer is to encode two sentence sequences into a vector representation with context information. The Word Pair Extraction Layer extracts multiple keyword pairs using the attention mechanism. The Denoising Layer first denoises each sentence, and then obtains the key semantic feature vector of the sentence. The Matching Layer uses two feature vectors to obtain interactive features. Finally, the Prediction Layer uses interactive features to predict sentence-semantic relationships. MKPM includes two tasks: WP-Task and SP-Task. WP-task regards each keyword pair as a pair of word semantic vectors, and then sends them to the Matching Layer and Prediction Layer to predict the relationship between two sentences. SP-task first obtains the semantic vectors of two sentences after noise reduction through the Denoising Layer, and then sends them to the Matching Layer and Prediction Layer to complete the classification. Then the Matching Layer and Prediction Layer are used to complete the prediction of the relationship between the two sentences.

#### 3.2.1 Word representation layer

The goal of this layer is to obtain the context representation of each word in the sequences  $P$  and  $Q$ , including the Word Embedding Layer and the Context Layer (Fig. 1). The Word Embedding Layer consists of three parts (for each word  $w$ ): 1. Word embedding: We use Word2vec [32]

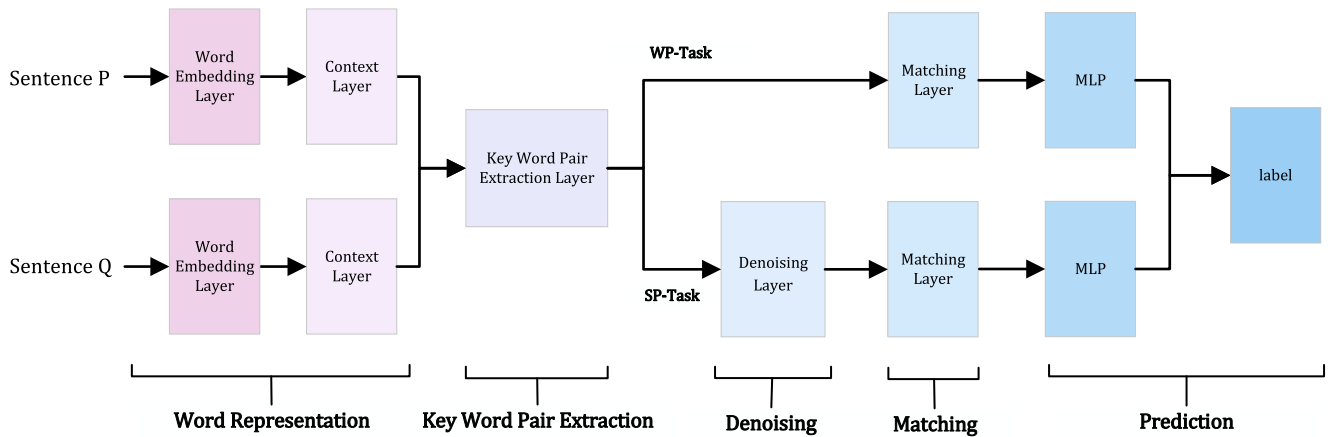


Fig. 1 Architecture of MKPM

and Glove [33] to pre-train the word vector as  $E1(w) = Word2vec(w)$  or  $Glove(w)$ . 2. Character embedding: In order to solve the OOV (out of vocabulary) problem, we need to use the features of characters to represent words. First, each character of a word is embedded as a character, and then the character sequence is sent to the LSTM to get the character representation of the word as  $E2(w) = Char-LSTM(w)$ . 3. Exact word matching [4]: If word  $w$  exists in the other sentence,  $E3(w)$  is marked as 1, as shown in (1), otherwise it is marked as 0. The three parts are cascaded together as a word embedding representation  $E = [E1(w); E2(w); E3(w)]$ . We get the word embedding of  $P$  and  $Q$  respectively as  $E^P = [E^{P1}, E^{P2}, \dots, E^{PM}]$  and  $E^Q = [E^{Q1}, E^{Q2}, \dots, E^{QN}]$ .

$$E3(w) = \begin{cases} 1 & \text{The word } w \text{ exists in the other sentence} \\ 0 & \text{Otherwise} \end{cases} \quad (1)$$

We use a 2-layer BiLSTM as the Context Layer to encode  $P$ , so that each time step of the sequence contains context semantic information. First, we calculate the hidden state of each time step of the first layer BiLSTM as shown in (2). Then we take the hidden state of the first layer as the input of the second layer, and calculate the hidden state of each time step of the BiLSTM of the second layer as shown in (3).

$$h^{P1} = BiLSTM_1(E^P) \quad (2)$$

$$h^{P2} = BiLSTM_2(h^{P1}) \quad (3)$$

Finally, we concatenate the hidden states of the two layers of the BiLSTM together as the context representation of  $P$  as shown in (4). We use the same network structure to encode  $Q$  as shown in (5).

$$h^P = [h^{P1}; h^{P2}] \quad (4)$$

$$h^Q = [h^{Q1}; h^{Q2}] \quad (5)$$

### 3.2.2 KWPE layer

The KWPE layer is the core part of our model as shown in Fig. 2. We propose a sentence-pair-based co-attention mechanism called *sp-attention*, which is used to calculate the attention value of sentence pairs  $(P, Q)$  for each word in  $P$  and  $Q$ . We think that for each word  $w_p$  in  $P$ , if the sentence pair  $(P, Q)$  has a large attention score on it, it should: 1. have rich semantic features, 2. be very important in  $P$ , and 3. have great influence on  $Q$ . Specifically, these three characteristics are respectively determined by  $P$ 's context features,  $h^P$ ,  $P$ 's attention score to  $P$ ,  $h^P h^{PT}$ , and  $Q$ 's attention score to  $P$ ,  $h^P h^{QT}$ . As shown in the  $P$ -module in Fig. 2, we use the *tanh* function to activate these three weighted sums, then transform them linearly, and finally use the *softmax* function to calculate the *sp-attention* weight  $\alpha^P$  of each word in  $P$ . Similarly, we use the  $Q$ -module to calculate  $\alpha^Q$ .  $\alpha^P$  and  $\alpha^Q$  are 1-dimensional vectors. We wish to choose the words with the largest *sp-attention* score from  $P$  and  $Q$  as the selected keyword pair  $(w_p, w_q)$ . However, this cannot easily be achieved and therefore the effect of the choice is approximated (see below). The detailed calculation is shown in (6) to (11).

$$m^P = \tanh(h^P h^{PT} W_{pp} + h^P h^{QT} W_{pq} + h^P W_p) W_d \quad (6)$$

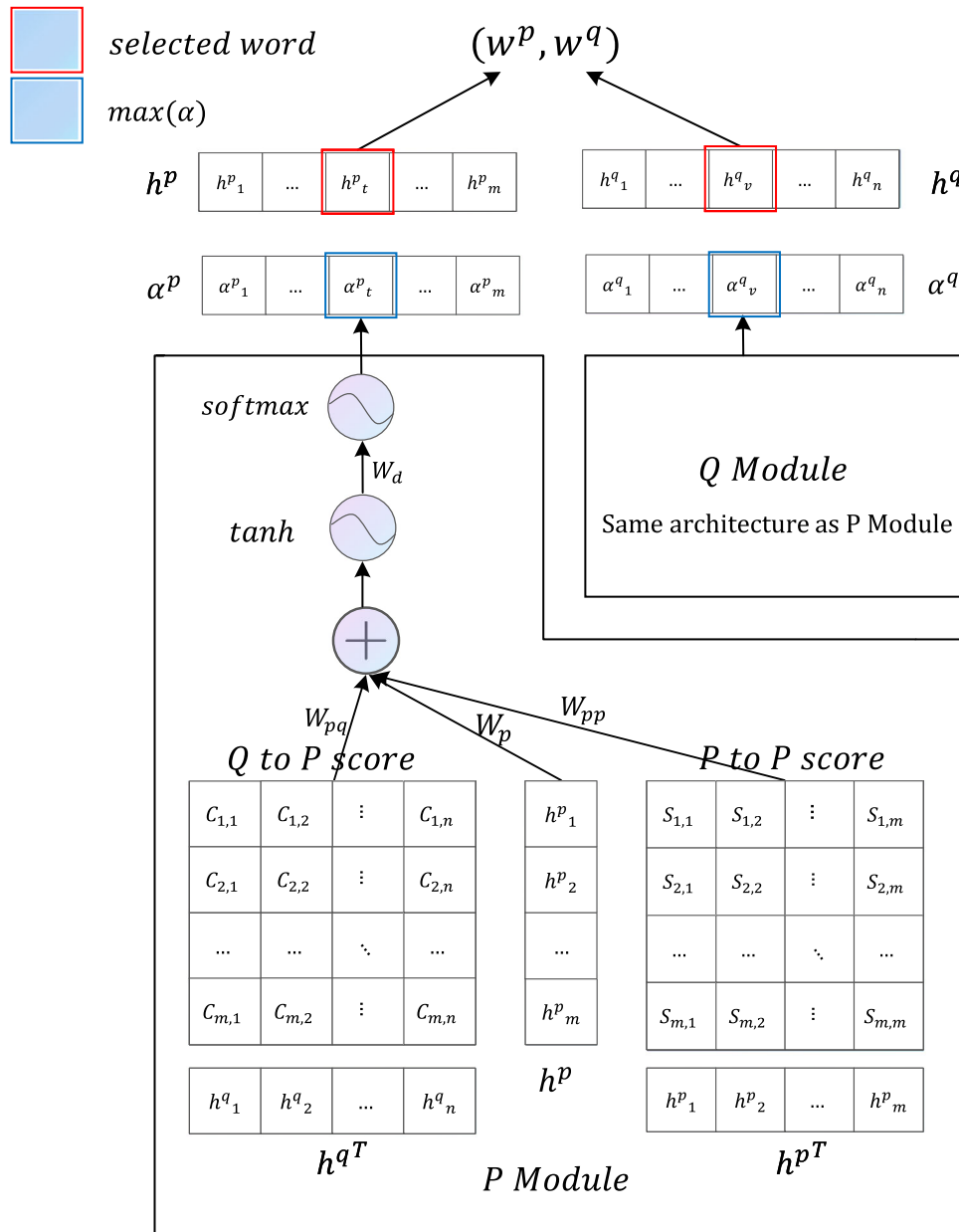
$$m^Q = \tanh(h^Q h^{QT} W_{qq} + h^Q h^{PT} W_{qp} + h^Q W_q) W_d \quad (7)$$

$$\alpha^P = \frac{\exp(m^P)}{\sum_{i=1}^M \exp(m_i^P)} \quad (8)$$

$$\alpha^Q = \frac{\exp(m^Q)}{\sum_{i=1}^N \exp(m_i^Q)} \quad (9)$$

$$w^P = h_t^P, (t = \text{index}(\max(\alpha^P))) \quad (10)$$

$$w^Q = h_v^Q, (v = \text{index}(\max(\alpha^Q))) \quad (11)$$



**Fig. 2** Architecture of KWPE Layer, where  $C_{i,j}$  represents the attention score between the  $i^{\text{th}}$  word of  $P$  and the  $j^{\text{th}}$  word of  $Q$ , and  $S_{i,j}$  represents the attention score between the  $i^{\text{th}}$  word of  $P$  and the  $j^{\text{th}}$  word of  $P$ . The P Module calculates the  $sp$ -attention score  $\alpha^p$  of the sentence pair  $(P, Q)$  to each word in  $P$ , and the Q Module calculates the  $sp$ -attention score  $\alpha^q$  of  $(P, Q)$  to each word in  $Q$ . The

blue box indicates finding the maximum  $sp$ -attention value, and the red box indicates selecting the word with the maximum  $sp$ -attention value. In fact, these operations are not easy to achieve and so they are approximated by sharpening the distribution of weights  $\alpha^p$  and then computing the sum of all  $h^p$ , weighted by their corresponding  $\alpha^p$  values. The same applies to  $\alpha^q$  and  $h^q$ . See the text for more details

where  $W_{pq}$ ,  $W_{pp}$ ,  $W_p$ ,  $W_{qp}$ ,  $W_q$ ,  $W_d$  are trainable parameters, and  $(w^p, w^q)$  is the selected keyword pair. The  $\max(*)$  function is not differentiable, which means its gradient cannot be calculated. Thus we cannot simply determine the maximum value in  $\alpha$ . Instead, we use a trick in Zhang et al. [31] to multiply  $m^p$  and  $m^q$  by a relatively large value  $\gamma$ , sharpening the distributions of  $\alpha^p$  and  $\alpha^q$ . After sharpening, the weight of the word with the largest

$sp$ -attention score in the sentence is very close to 1, while the weight of all other words is very close to 0, that is,  $\alpha^p$  and  $\alpha^q$  are very close to a one-hot distribution. Then for sentence  $P$ , the context vectors  $h^p$  for the words in  $P$  are weighted by their corresponding values in the sharpened  $\alpha^p$  and then summed to obtain the approximate representation of the keyword  $w$  which has the maximum  $sp$ -attention score. The same operation is performed on the context

vectors  $h^q$  for sentence  $Q$ , using the sharpened  $\alpha^q$ . The calculation process is shown in (12) to (15).

$$\alpha^p = \frac{\exp(m^p \gamma)}{\sum_{i=1}^M \exp(m_i^p \gamma)} \quad (12)$$

$$\alpha^q = \frac{\exp(m^q \gamma)}{\sum_{i=1}^N \exp(m_i^q \gamma)} \quad (13)$$

$$w^p \approx \sum_{i=1}^M \alpha_i^p h_i^p \quad (14)$$

$$w^q \approx \sum_{i=1}^N \alpha_i^q h_i^q \quad (15)$$

However, sharpening  $\alpha^p$  and  $\alpha^q$  at the beginning of training may cause the model either not to converge, or to converge to a suboptimal region. So we set up an annealing schedule: In the process of training, we increase the value of  $\gamma$ , as shown in (16).

$$\gamma_i = \min((i + 1)/3, \gamma_{\max}) \quad (16)$$

where  $i$  is the epoch of training,  $\gamma_{\max}$  is the maximum value of  $\gamma$ .

In order to capture the semantic relationship between two sentences at different levels, we need to extract multiple keyword pairs. By calculating attention  $K$  times, i.e. each time the model learns different parameters, we can obtain multiple keyword pairs. We obtain the semantic information of these keyword pairs through WP-Task and SP-Task.

### 3.2.3 Denoising layer

In order to enable SP-Task to better understand the key semantic information of the sentence, we only use the keyword sequence of the sentence to represent it, so we propose the Denoising Layer, as in Fig. 3. As described in the previous section, we use the approximate keyword pairs extracted by the KWPE layer, and mark the selected keyword pairs with a red box in the original sequence. We take these marked words as a new sequence, and then use the BiLSTM network to encode it. Finally, we cascade

the vector in the last time-step of the BiLSTM network to construct the keyword semantic feature vectors  $V^p$  and  $V^q$  of sentences  $P$  and  $Q$ .

### 3.2.4 Matching layer

This layer mainly obtains the interaction information of the two input vectors  $V_1$  and  $V_2$ . We use the processing method of Kim [29] to calculate the interaction between two semantic vectors, expecting to obtain local differences or conflict information between the vectors to infer the relationship between the two sentences. The interaction information is extracted in both WP-Task and SP-Task:

1. WP-Task: We use the  $K$  word pairs selected by the KWPE layer as semantic vectors to allow the model to understand the semantic relationship at word granularity. The interaction feature vector  $F_{wpk}$  of each set of keyword pairs is obtained as shown in (17).

$$F_{wpk} = [w_k^p, w_k^q, w_k^p + w_k^q, w_k^p - w_k^q, |w_k^p - w_k^q|], \quad k = 1, 2, \dots, K \quad (17)$$

2. SP-Task: We take the vector representation of the sentence extracted by the Denoising Layer as the semantic vector input, and let the model understand the semantic relationship at sentence granularity. The interactive feature vector  $F_{sp}$  of the two sentences is obtained as shown in (18).

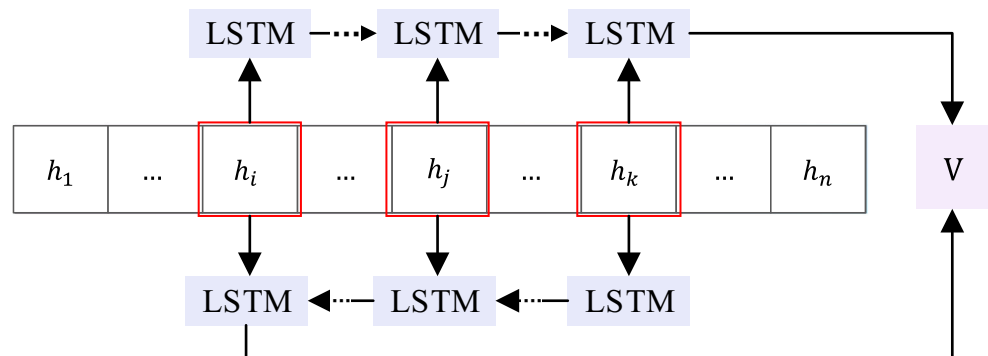
$$F_{sp} = [V^p, V^q, V^p + V^q, V^p - V^q, |V^p - V^q|] \quad (18)$$

The reason why we design these two tasks with different granularity is that we hope the network can learn different levels of semantic information. By combining these two tasks, we can improve the prediction accuracy of the model.

### 3.2.5 Prediction layer

We feed the interaction feature vector into a multi-layer perceptron (MLP) to predict the relationship between two sentences. The MLP has two fully-connected hidden layers activated by ReLU, and an output layer activated

**Fig. 3** Architecture of Denoising Layer





by softmax. We first feed the interaction vectors of  $K$  word pairs extracted by WP-Task into  $K$  different MLPs for classification, and then take the average value as the prediction result of WP-Task. Then the sentence semantic interaction vector extracted by SP-Task is sent to the MLP to obtain the prediction result of SP-Task. We define the contribution of SP-Task to the model as  $\beta$ , and the contribution of WP-Task to the model as  $1 - \beta$ , where  $\beta$  is an adjustable super parameter. The prediction result of MKPM is shown in (19), and the model is trained by minimizing the cross-entropy loss.

$$\text{Predict} = \beta \times MLP^{sp}(F_{sp}) + (1 - \beta) \times \frac{1}{K} \sum_{k=1}^K MLP_k^{wp}(F_{wpk}) \quad (19)$$

## 4 Experiments and results

In this section, we first describe four benchmark sentence matching datasets, and then introduce our experiments and results analysis.

### 4.1 Datasets

We briefly introduce the datasets used in our paper:

- Large-scale Chinese Question Matching Corpus (LCQMC) [13]: This is an open-domain Chinese dataset collected from Baidu Knowledge (a popular Chinese community QA website). Its goal is to determine whether the semantics of two questions are the same, and it focuses on intent matching rather than paraphrase. LCQMC includes 239k sentence pairs for training, 8.4k pairs for validation, and 12.5k pairs for testing.
- Bank Question (BQ) corpus [18]: It contains a 1-year customer service log from an online bank. BQ is a dataset for Sentence Semantic Equivalence Recognition (SSEI) and it is the largest manually annotated Chinese dataset in the banking field. BQ includes 100k training sentence pairs, 10k validation pairs, and 10k test pairs.
- Quora Question Pair (QQP): This is an English dataset of question matching published by Quora, whose purpose is to determine whether the semantics of two questions are the same. QQP includes 360k training sentence pairs and 40k validation pairs. We split the QQP dataset into training set, validation set and test set according to [1]'s method.
- The Stanford Natural Language Inference (SNLI) Corpus [19]: It is a collection of human-written English sentence pairs. These sentence pairs are manually annotated with entailment, contradiction, and neutral

labels, and support natural language inference tasks. SNLI includes 550k sentence pairs for training, 10k pairs for validation, and 10k pairs for testing.

### 4.2 Implementation details

We use 300d Word2vec [34] Chinese word vectors pre-trained on the Baidu Encyclopedia corpus and 300d GloVe [33] English word vectors pre-trained on the 840B Common Crawl corpus to initialize word embeddings, and randomly initialize out-of-vocabulary word embeddings. We also randomly initialize character embeddings and use the LSTM to extract the character representation of the word. In addition, we use Batch Normalization [35] to speed up model training and prevent overfitting. The hidden layers of the LSTM all have 128 units. We apply the RMSProp optimizer with an initial learning rate of 0.001. Except for the embedding matrix, all weights are constrained by L2 regularization with a regularization constant  $\lambda = 10^{-6}$ . The settings of each dataset are shown in Table 1, where Sentence length is the length of our intercept sequence, Word embedding is the training method and dimension of the pre-trained word vectors, Word length is the character length of each word, Word dim is the initial vector dimension of each word, Char dim is the character embedding dimension of each word, and OOV is the proportion of out-of-vocabulary words. We choose the model that works best on the validation sets, and then evaluate it on the test sets.

### 4.3 Baselines

We briefly introduce the Baselines as follows:

1. Text Convolutional Neural Network (Text-CNN) [20] is a CNN model for sentence classification. The model represents each sentence as an embedding matrix, and then uses the convolutional neural network to classify the sentence.
2. Siamese Long Short-term Memory (Siamese-LSTM) is a variant of RNN which considers both long and short dependencies in context, working both forward and backward. Mueller et al. and Varior et al. [23, 24, 36] use the same LSTM to encode sentences into sentence vectors in both forward and backward directions, and then match the two sentence vectors.
3. Lexical Decomposition and Composition (L.D.C.) [4] is a model to take into account both the similarities and dissimilarities by decomposing and composing lexical semantics over sentences. The model represents each word as a vector, and calculates the semantic matching vector of each word according to all the words in the other sentence.

**Table 1** Dataset parameter settings

Dataset	Sentence length	Word embedding	Word length	Word dim	Char dim	OOV
LCQMC	25	300d Word2vec	5	16	50	10%
BQ	15	300d Word2vec	5	16	50	13%
QQP	25	300d Glove	16	16	50	25%
SNLI	30	300d Glove	16	16	50	10%

4. Pretrained Decomposable Attention Character n-gram (pt-DecAttchar.c) [37] is a decomposable attention model, extended with character n-gram embeddings and noisy pretraining for the task of question paraphrase identification.
5. Compare Propagate Alignment-Factorized Encoders (CAFE) [38] is a new deep learning architecture for Natural Language Inference; it compares and compresses aligned pairs, and then propagates to the upper layer to enhance representation learning.
6. Gumbel Tree Long Short-term Memory (Gumbel Tree-LSTM) [39] is a novel tree-structured LSTM that learns how to compose task-specific tree structures efficiently, using only plain-text data.
7. Distance-based Self-Attention Network (Distance-based SAN) [40] determines the word distance by using a simple distance mask in order to model the local dependency without losing the ability to model global dependency.
8. Bilateral Multi-Perspective Matching (BiMPM) [1] is a model with good performance for sentence matching. The model uses a BiLSTM to learn sentence representations, matches sentences from two directions and multiple perspectives, aggregates the matching results with a BiLSTM, and finally makes predictions with fully-connected layers.
9. Densely Interactive Inference Network (DIIN) [11] achieves a high-level understanding of sentence pairs by extracting semantic features hierarchically from the interaction space.
10. Dynamic Re-read Network (DRr-Net) [31] pays close attention to the important content of the sentence in each step, by re-reading important words to better understand the semantics of the sentence.
11. Densely-connected Recurrent and Co-attentive neural Network (DRCN) [29] uses within each layer the attention features and hidden features of all previous recursive layers to achieve accurate understanding of sentences.

#### 4.4 Results and analysis

We first evaluate the performance of the model on the two Chinese paraphrase identification datasets LCQMC and BQ; next, in order to prove that our method is suitable

for different languages, we apply the model to the English dataset QQP. At the same time, in order to demonstrate that our method is suitable for different sentence matching tasks, we evaluate it on the natural language inference dataset SNLI. We conduct an ablation study to prove the effectiveness of SP-Task, WP-Task, the KWPE layer, and the annealing schedule. Then, we explore the impact of the number  $K$  of extracted keyword pairs on the model's performance. Finally, we qualitatively evaluate the extracted keyword pairs to verify the validity and accuracy of the model.

##### 4.4.1 Experiment results

We show the experimental results of our model on the four datasets, LCQMC, BQ, QQP and SNLI, see Tables 2, 3 and 4.

Table 2 shows the performance of our model on the two Chinese datasets LCQMC and BQ, compared with the baselines, where (1)-(4) are from [13, 18], and (5) is our implementation of the DRCN model in [29], using the same word embeddings to obtain the model's performance. The results show that, for the two datasets, our model is 3.3% and 2.2% higher than the sentence matching BiMPM model and it is 0.8% and 0.8% higher than our implementation of the DRCN model.

Table 3 shows the performance of our model on the English dataset QQP, compared with the baselines, where (1)-(5) are from [11], (6) is from [31], (7) is from [29], and (8) is our implementation of the DRCN model in [29]. Experimental results show that our model is superior to the general sentence matching models (1)-(5), and is also better than BiMPM and DIIN. Rows (6)-(7) show that our model is 0.1% lower

**Table 2** MKPM performance (accuracy) on LCQMC & BQ

Model	LCQMC	BQ
(1) Text-CNN	72.8	68.52
(2) BiLSTM	76.1	73.51
(3) BiMPM	83.4	81.85
(4) DIIN	-	81.41
(5) DRCN	85.9	83.3
MKPM	<b>86.71</b>	<b>84.11</b>

Best results are in bold



**Table 3** MKPM performance (accuracy) on QQP

Model	QQP
(1) Siamese-LSTM	82.58
(2) L.D.C.	85.55
(3) BiMPM	88.17
(4) pt-DecAttchar.c	88.40
(5) DIIN	89.06
(6) DRr-Net	89.75
(7) DRCN	<b>90.15</b>
(8) our DRCN	89.6
MKPM	89.66

Best results is in bold

in performance than DRr-Net and 0.5% lower than DRCN. In row (8), our implementation of DRCN achieves 89.6% accuracy, which is close to the author's score. Table 4 shows the results on the English dataset SNLI, where (1)-(5) are from [31]. Here, our model performs better than the general natural language inference models (1)-(5). Row (5) shows that our model is 0.5% higher than the best previous model DRr-Net, and our model parameters (1.8m) are only half of DRr-Net (3.5m) and one third of DRCN (5.6m). Although our model does not perform as well as DRCN and DRr-Net on the QQP dataset, it has far fewer parameters than them. Generally, these two experiments show that our model not only adapts to different languages, but also has high performance on different sentence matching tasks, while using fewer parameters.

#### 4.4.2 Ablation study

We conduct ablation studies on the experiments working with the LCQMC and BQ datasets, aiming to verify the effectiveness of the 2-layer BiLSTM representation, the WP-Task, the SP-Task, the KWPE layer and the annealing schedule. The results are shown in Table 5, where we set the keyword pairs parameter  $K = 5$ . First, we study the impact of the number of BiLSTM layers on the performance of

**Table 4** MKPM performance (parameters and accuracy) on SNLI

Model	Params	Accuracy
(1) CAFE	3.7m	85.9
(2) Gumbel TreeLSTM	2.9m	86.0
(3) Distance-based SAN	4.7m	86.3
(4) DRCN	5.6m	86.5
(5) DRr-Net	3.5m	87.7
MKPM	<b>1.8m</b>	<b>88.2</b>

Best results are in bold

the model when representing the context. We remove the 2-layer BiLSTM and use only a single layer BiLSTM to encode the sentence, as shown in row (1) of the table. We can see that if we only use a 1-layer BiLSTM to encode sentences, the model performance decreases by 0.2%. Next, we verify the impact of the WP-Task and SP-Task on model performance. We remove one of the tasks, as shown in rows (2) and (3) in the table. Whichever task is removed, the performance of the model declines. In particular, when WP-Task is removed, the performance drops by about 0.7%, so the importance of WP-Task is self-evident. Then we study the contribution of the two tasks to the model. We set  $\beta$  to 0.9, 0.7, 0.5, 0.3, 0.1 as shown in rows (4)-(8) of the table. We see that when  $\beta$  is 0.7 and 0.5, the model achieves the highest performance on the two datasets. Therefore, we conclude that the performance of the two tasks WP-Task and SP-Task is mutually reinforcing, and the model achieves the highest performance when  $\beta$  is around 0.5 to 0.7. Finally, in order to verify the contribution of the KWPE layer to the model, we remove it in row (9). Performance is reduced by 2.7% on LCQMC and 2.6% on BQ, indicating that the KWPE layer is irreplaceable. To verify the effectiveness of the annealing schedule to the model, we remove it in row (10), and find that the performance is slightly decreased, which proves the effectiveness of the annealing schedule.

#### 4.4.3 Analysis of the number of keyword pairs

We set the value of  $\beta$  on LCQMC and BQ to 0.7 and 0.5 respectively, and set the word pair parameter  $K = 2, 3, 4, 5, 6, 7, 8$ . We study the effect of the number of keyword pairs on the model's performance. The results are shown in Fig. 4. From the figure, it can be seen that if the word pair parameter  $K$  is too large or too small, the performance of the model will be reduced. When  $K = 6$  on the LCQMC data set, the model performance reaches the highest value

**Table 5** Ablation study performance (accuracy) on LCQMC & BQ

Model	LCQMC	BQ
MKPM	86.71	84.11
(1) 1-layer BiLSTM	86.44	83.86
(2) $\beta = 1$ no (WP-Task)	86.02	83.21
(3) $\beta = 0$ no (SP-Task)	86.35	83.62
(4) $\beta = 0.9$	86.43	83.41
(5) $\beta = 0.7$	86.71	83.80
(6) $\beta = 0.5$	86.63	84.11
(7) $\beta = 0.3$	86.56	83.92
(8) $\beta = 0.1$	86.52	83.73
(9) w/o KWPE layer	84.03	81.47
(10) w/o Annealing schedule	86.57	83.93

of 86.71%, and when  $K = 5$  on the BQ data set, the model performance reaches its highest value of 84.11%, so for these two data sets, a value of 5 or 6 for  $K$  is most appropriate. From the experimental results, we can conclude that when the value of the word pair  $K$  is small, the model cannot learn the full semantic relationship of the sentence by a small number of word pairs, resulting in low performance. When the value of the word pair  $K$  is too large, the word pair extraction may include all the words in the sentence, which will contain redundant semantic information, so it will also cause the model performance to decline.

#### 4.4.4 Qualitative evaluation

We select four sentences to visualize the prediction results and extract keyword pairs from the LCQMC test set as shown in Table 6, where “ $T \rightarrow P$ ” means “ $True\ label \rightarrow Predict\ label$ ”. For the sentence pair “如何在网页上下载音频 (En): How to download audio on the web”, “怎么下载这个网页上的音频? (En): How to download audio on this page?”, the model extracts five keyword pairs “(如何,怎么), (上,网页), (在,这个), (网页,网页), (音频,音频) (En): (How, How), (on, page), (on, this), (web, page), (audio, audio)”. Through these word pairs, we can see that the objects described by the two sentences are audio on the web page, and we have not found a word pair that can indicate that the semantic relationship between the two sentences is not the same; so the model thinks that these two sentences have the same semantics, and hence

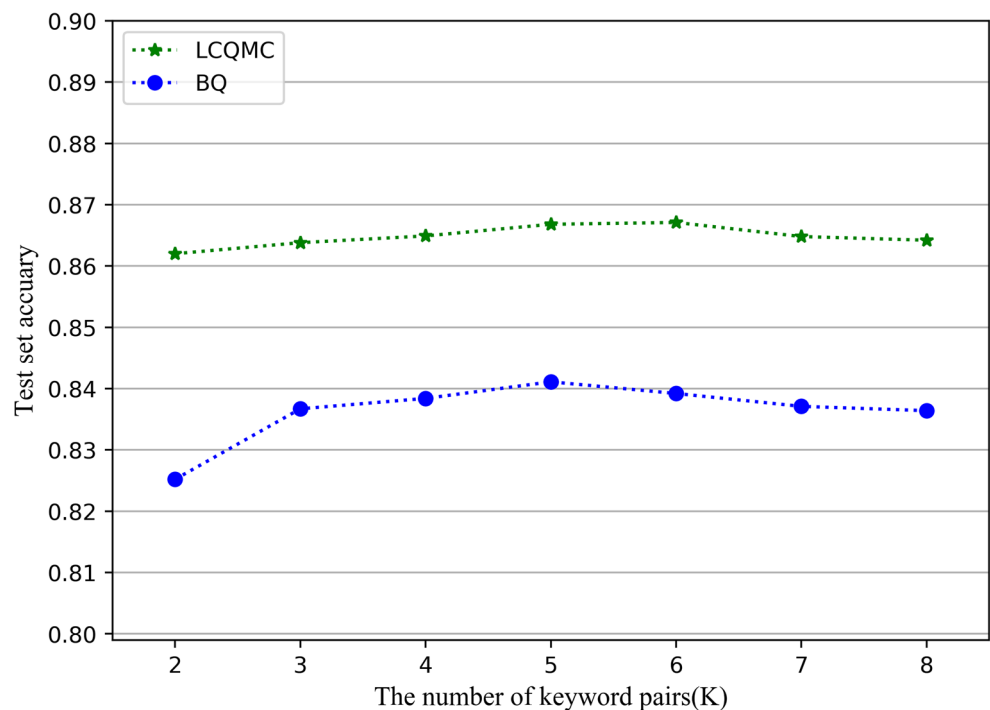
the label is predicted to be 1. For the sentence pair “桂林山水在哪个省 (En): Which province is Guilin’s scenic spot in?”, “桂林山水在哪个市 (En): Which city is Guilin’s scenic spot in?”, the model extracts five keyword pairs “(省,市), (省,市), (省,市), (省,市), (省,市) (En): (province, city), (province, city), (province, city), (province, city), (province, city)”. We find that these are five identical word pairs, and the word pair “(province, city)” means that the location information expressed in the first sentence is province, while in the second sentence it is city; when the model finds a keyword pair with different semantics, the label is predicted to be 0. From the above analysis, we see that when the model predicts the semantic relationship between two sentences, it tries to find keyword pairs that represent differences between them. If it cannot find such keyword pairs, it will predict the label as 1. Conversely, if it can find a keyword pair with different semantics, it will directly predict the label as 0.

#### 4.5 Discussion

The experimental results show that our model uses WP-Task to understand the semantic relationship at the word level, and SP-Task to understand the semantic relationship at the sentence level. Through the integration of the two tasks, our model can understand sentences more accurately by means of the two granularities, words and sentences.

Our method achieves the state-of-the-art performance on two Chinese question matching datasets LCQMC and BQ. Experiments on the English dataset QQP for question

**Fig. 4** Impact of the Number of Keyword Pairs,  $K$ , on Accuracy



**Table 6** Keyword-pair visualization results

	<i>Sentence Pair</i>	<i>Word Pairs</i>	$T \rightarrow P$
s1	如何在网页上下载音频 (En): How to download audio on the web?	(如何, 怎么) (上, 网页) (在, 这个) (网页, 网页) (音频, 音频)	1 $\rightarrow$ 1
s2	怎么下载这个网页上的音频? (En): How to download audio on this page?	(En): (How, How) (on, page) (on, this) (web, page) (audio, audio)	
s1	嚼口香糖能减肥吗 (En): Can you lose weight by chewing gum?	(能, 会) (减肥, 减肥) (嚼, 嚼) (口香糖, 口香糖) (减肥, 减肥)	1 $\rightarrow$ 1
s2	嚼口香糖会减肥吗? (En): Will you lose weight by chewing gum?	(En): (Can, will) (lose weight, lose weight) (chewing, chewing) (gum, gum) (lose weight, lose weight)	
s1	桂林山水在哪个省 (En): Which province is Guilin's scenic spot in?	(省, 市) (省, 市) (省, 市) (省, 市) (省, 市)	0 $\rightarrow$ 0
s2	桂林山水在哪个市 (En): Which city is Guilin's scenic spot in?	(En): (province, city) (province, city) (province, city) (province, city) (province, city)	
s1	你喜欢哪个节日 (En): Which festival do you like?	(哪个, 我) (哪个, 的) (你, 我) (你, 我) (节日, 节日)	0 $\rightarrow$ 0
s2	我喜欢的节日 (En): (What is) my favorite festival?	(En): (which, my) (which, of) (you, my) (you, my) (festival, festival)	

matching and the natural language inference dataset SNLI show that our model not only adapts to different languages, but can also be applied to different sentence matching tasks. At the same time, our model has fewer parameters. However, because the model is keyword-based, the quality of words has a great impact on its performance. In summary, the model has the following shortcomings: 1. It is sensitive to OOV words. In QQP, for example, OOVs account for 25%, while in LCQMC and SNLI they only comprise 10%. In consequence, our model performs better on LCQMC and SNLI than on QQP. 2. Differences between Chinese and English words lead to the performance of our model being better on the Chinese dataset than it is on the English dataset. In English, a character only represents the pronunciation, while a word not only includes semantic features, but also carries complex grammatical information. In consequence, it is difficult to capture the key semantics of an English sentence through keywords alone. In Chinese, by contrast, a character includes both pronunciation and semantic features, while a word is composed of multiple characters, increasing still more the information being conveyed. Hence, it is easier to understand the key semantics of a Chinese sentence through keywords. 3. Our model contains two hyper-parameters  $K$  and  $\beta$ , which need to be adjusted.

## 5 Conclusion

In this paper, we propose a sentence matching method called MKPM. We only use multiple keyword pairs to carry out the task. We propose for the first time that such keyword pairs are used to represent multi-level semantic relationships between sentences. Experimental results show that our model can achieve state-of-the-art performance in several sentence matching tasks. In addition, compared with the most advanced methods, our model is simpler and has fewer parameters. In order to achieve a better performance, our future work can be divided into three parts:

1. Let the model automatically learn better hyper-parameters.
2. Study networks with Dense structure to make the model understand sentences more deeply.
3. Make the model more adaptable to OOV words and different languages.
4. Improve the annealing schedule for model training.

**Acknowledgements** The authors would like to thank the anonymous reviewers for their constructive comments. This work was supported by the Shaanxi Provincial Science and Technology Department of China (No. 2019ZDLGY03-10) and National Natural Science Foundation of China (Grant no. 61877050).

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References

- Wang Z., Hamza W., Florian R. Bilateral multi-perspective matching for natural language sentences, arXiv:1702.03814
- Madnani N., Tetreault J., Chodorow M. (2012) Re-examining machine translation metrics for paraphrase identification. In: Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics. Human Language Technologies, Association for Computational Linguistics, pp 182–190
- Chen Q., Zhu X., Ling Z., Wei S., Jiang H., Inkpen D. Enhanced lstm for natural language inference, arXiv:1609.06038
- Wang Z., Mi H., Ittycheriah A. Sentence similarity learning by lexical decomposition and composition, arXiv:1602.07019
- Esposito M., Damiano E., Minutolo A., De Pietro G., Fujita H. (2020) Hybrid query expansion using lexical resources and word embeddings for sentence retrieval in question answering. *Inf Sci* 514:88–105
- Liu P., Qiu X., Chen J., Huang X.-J. (2016) Deep fusion lstms for text semantic matching. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp 1034–1043
- Pota M., Marulli F., Esposito M., De Pietro G., Fujita H. (2019) Multilingual pos tagging by a composite deep architecture based on character-level features and on-the-fly enriched word embeddings. *Knowl-Based Syst* 164:309–323
- Pota M., Esposito M., Pietro G. D., Fujita H. (2020) Best practices of convolutional neural networks for question classification. *Appl Sci* 10(14):4710
- Catelli R., Casola V., De Pietro G., Fujita H., Esposito M. (2021) Combining contextualized word representation and sub-document level analysis through bi-lstm+ crf architecture for clinical de-identification. *Knowl-Based Syst* 213:106649
- Catelli R., Gargiulo F., Casola V., De Pietro G., Fujita H., Esposito M. A novel covid-19 data set and an effective deep learning approach for the de-identification of italian medical records. *IEEE Access*
- Gong Y., Luo H., Zhang J. Natural language inference over interaction space, arXiv:1709.04348
- Yin W., Schütze H., Xiang B., Zhou B. (2016) ABCNN: Attention-Based convolutional neural network for modeling sentence pairs. *Trans Assoc Comput Linguist* 4:259–272
- Liu X., Chen Q., Deng C., Zeng H., Chen J., Li D., Tang B. (2018) LCQMC: A large-scale Chinese question matching corpus. In: Proceedings of the 27th International Conference on Computational Linguistics, pp 1952–1962
- MacCartney B., Manning C. D. (2009) Natural language inference. *Citeseer*
- Bilotti M. W., Ogilvie P., Callan J., Nyberg E. (2007) Structured retrieval for question answering. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp 351–358
- Wang M., Smith N. A., Mitamura T. (2007) What is the Jeopardy model? a quasi-synchronous grammar for QA. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pp 22–32
- Moldovan D., Clark C., Harabagiu S., Hodges D. (2007) Cogex: a semantically and contextually enriched logic prover for question answering. *J Appl Log* 5(1):49–69
- Chen J., Chen Q., Liu X., Yang H., Lu D., Tang B. (2018) The BQ corpus: A large-scale domain-specific Chinese corpus for sentence semantic equivalence identification. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp 4946–4951
- Bowman S. R., Angeli G., Potts C., Manning C. D. A large annotated corpus for learning natural language inference. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics
- Kim Y. Convolutional neural networks for sentence classification, arXiv:1408.5882
- Dey R., Salemt F. M. (2017) Gate-variants of gated recurrent unit (GRU) neural networks. In: 2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS). IEEE, pp 1597–1600
- Greff K., Srivastava R. K., Koutník J., Steunebrink B. R., Schmidhuber J. (2016) Lstm: A search space odyssey. *IEEE Trans Neural Netw Learn Syst* 28(10):2222–2232
- Mueller J., Thyagarajan A. (2016) Siamese recurrent architectures for learning sentence similarity. In: Thirtieth AAAI conference on artificial intelligence
- Variar R. R., Shuai B., Lu J., Xu D., Wang G. (2016) A siamese long short-term memory architecture for human re-identification. In: European Conference on Computer Vision. Springer, pp 135–153
- Nie Y., Bansal M. Shortcut-stacked sentence encoders for multi-domain inference, arXiv:1708.02312
- Shen T., Zhou T., Long G., Jiang J., Wang S., Zhang C. Reinforced self-attention network: A hybrid of hard and soft attention for sequence modeling, arXiv:1801.10296
- Huang Q., Bu J., Xie W., Yang S., Wu W., Liu L. (2019) Multi-task sentence encoding model for semantic retrieval in question answering systems. In: 2019 international joint conference on neural networks (IJCNN). IEEE, pp 1–8
- Wang S., Jiang J. A compare-aggregate model for matching text sequences, arXiv:1611.01747
- Kim S., Kang I., Kwak N. (2019) Semantic sentence matching with densely-connected recurrent and co-attentive information. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol 33, pp 6586–6593
- Zheng G., Mukherjee S., Dong X. L., Li F. (2018) Opentag: Open attribute value extraction from product profiles. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp 1049–1058
- Zhang K., Lv G., Wang L., Wu L., Chen E., Wu F., Xie X. (2019) DRr-Net: Dynamic re-read network for sentence semantic matching. In: Proceedings of the AAAI Conference on Artificial Intelligence, Vol 33, pp 7442–7449
- Mikolov T., Chen K., Corrado G., Dean J. Efficient estimation of word representations in vector space, arXiv:1301.3781
- Pennington J., Socher R., Manning C. D. (2014) Glove: Global vectors for word representation. In: Proceedings of the



- 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp 1532–1543
34. Li S., Zhao Z., Hu R., Li W., Liu T., Du X. Analogical reasoning on Chinese morphological and semantic relations, arXiv:[1805.06504](#)
  35. Ioffe S., Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift, arXiv:[1502.03167](#)
  36. Wang Z., Mi H., Ittycheriah A. Semi-supervised clustering for short text via deep representation learning, arXiv:[1602.06797](#)
  37. Tomar G. S., Duque T., Täckström O., Uszkoreit J., Das D. Neural paraphrase identification of questions with noisy pretraining, arXiv:[1704.04565](#)
  38. Tay Y., Tuan L. A., Hui S. C. A compare-propagate architecture with alignment factorization for natural language inference. arXiv:[1801.00102](#)
  39. Choi J., Yoo K. M., Lee S-g Learning to compose task-specific tree structures, arXiv:[1707.02786](#)
  40. Im J., Cho S. Distance-based self-attention network for natural language inference, arXiv:[1712.02047](#)

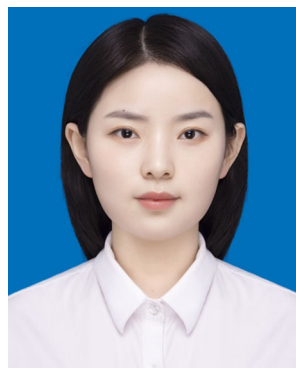
**Publisher's note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Xin Lu** received a B.S. in Computer Science from Northwest University China in 2018. He is studying for an M.S. degree at Northwest University. His research is in the areas of Natural Language Understanding (NLU), including sentence representation, sentence matching and other NLU tasks.



**Yao Deng** received a B.S. in Computer Science from Northwest University China in 2019. She is studying for an M.S. degree at Northwest University. Her research is in the areas of Natural Language Processing, including sentence matching and sentence representation. She also studies the application of GNNs to sentence matching tasks.



**Ting Sun** received a B.S. in Computer Science from Northwest University China in 2018. She is studying for an M.S. degree at Northwest University. Her research is in the areas of Machine learning and Blockchain.



**Yi Gao** received a B.S. in Computer Science from Northwest University China in 2017. She is studying for an M.S. degree at Northwest University. Her research is in the areas of Neural Networks and Natural Language Processing. In addition to sentiment classification, she has applied deep learning to student performance prediction.



**Jun Feng** received a Ph.D. from City University of Hong Kong in 2006. She is a Professor in the School of Information Science and Technology at Northwest University. Her research areas include pattern recognition and machine learning, especially in the fields of medical imaging analysis and intelligent education.

Recent projects have included medical image analysis with deep learning, and intelligent education based on

AI and Brain-Human Interaction.

She has reviewed for many journals, including TSP, JIVP, MTAP, JDIM, CJC, JCAD, OPE, and INFPHY. Conferences she has reviewed for include IEEE-VR, MICCAI, SIGCSE, IWCSE, and CompEd. She is a member of IEEE and ACM, and is co-author of 132 articles and co-editor of three books.





**Xia Sun** received a Ph.D. from Xian Jiaotong University, China, in 2006. She is a Professor in the School of Information Science and Technology at Northwest University, Xian, China. Her current research interests include natural language processing and machine learning.

Recent projects have included causality extraction from educational data, and relationship extraction from bioinformatics text.

She has served as a reviewer for IEEE Transactions on Industrial Informatics, IEEE Transactions on Neural Networks and Learning Systems and Chinese Journal of Electronics. She has also reviewed for the IEEE International Conference on Computational Science and Engineering. She is the co-author of 40 articles and is editor or co-editor of four books.









**Richard Sutcliffe** received a Ph.D. from University of Essex in 1989. He is an Associate Professor at Northwest University China. His research interests are in the areas of Natural Language Processing, Information Retrieval and Music Information Retrieval.

Recent projects have included persuasive conversational agents, public sector message classification, analysis of classical music texts, and personality and translation ability.

He has reviewed for Artificial Intelligence Review, Computational Linguistics, Computers and the Humanities, Information Processing and Management, Information Retrieval Journal, Journal of Natural Language Engineering, Journal Traitement Automatique des Langues. Conferences he has reviewed for include ACL, CIKM, COLING, IJCNLP, LREC, NAACL-HLT, and SIGIR. He is the co-author of 108 articles and is co-editor of three books and ten conference proceedings.

## Affiliations

Xin Lu<sup>1</sup>  · Yao Deng<sup>1</sup>  · Ting Sun<sup>1</sup>  · Yi Gao<sup>1</sup> · Jun Feng<sup>1</sup>  · Xia Sun<sup>1</sup>  · Richard Sutcliffe<sup>1,2</sup> 

Xin Lu  
rushin@stumail.nwu.edu.cn

Yao Deng  
dengyao@stumail.nwu.edu.cn

Ting Sun  
sunting@stumail.nwu.edu.cn

Yi Gao  
gaoyi@stumail.nwu.edu.cn

Jun Feng  
fengjun@nwu.edu.cn

<sup>1</sup> School of Information Science and Technology, Northwest University, Xi'an, 710127, China

<sup>2</sup> School of Computer Science and Electronic Engineering, University of Essex, Wivenhoe Park, Colchester, CO4 3SQ, UK